Claims: I claim:

SUB A5

1. A method for storing a plurality of parallel data element sequences comprising the steps of:

   (a) creating a dictionary of unique values for each of said data element sequences, whereby each dictionary associates a numeric index with each unique value in the corresponding sequence;

   (b) forming an n-ary tree with leaf and interior nodes where:

      (1) each leaf node corresponds to one of said dictionaries,

      (2) each interior node associates a numeric index with tuples of numeric indexes from other subordinate leaf or interior nodes, and

      (3) interior nodes may store sequences of unique, mutually-consecutive tuples separately from the other tuples.

2. The method of claim 1, whereby each unique value of a leaf node and each unique tuple of an interior node is associated with a count of the number of times that value or implied tuple of values occurred in the parallel data element sequences.

3. The method of claim 1, further including a means for efficiently processing a subset of a tree's leaves, comprising the following steps:

   (a) the definition of a gate field in interior nodes,
   (b) setting each of said gate fields' values, to indicate which of the corresponding interior node's branches lead to leaf nodes in said subset

(c) following paths that lead to said leaves, and

(d) processing the leaves encountered.

4. The method of claim 1, further including selectively disabling separate storage of tuple runs at certain interior nodes.

5. The method of claim 1, further including the method for arranging said n-ary tree comprising the steps of:

(a) defining a problem space consisting of:

(1) a set of states such that each state contains a set of leaves and zero or more interior nodes, each with two or more other nodes as children,

(2) a value function, giving a numeric ranking of the value of any state's design

(b) defining one or more operators that transform one state to another, and

(c) searching the problem space, starting from an initial state and applying operators to move to other states until a state with an acceptable design is reached.

6. A method for storing a plurality of parallel data element sequences comprising the steps of:

(a) creating a dictionary of unique values for each of said data element sequences, whereby each dictionary associates a numeric index with each unique value in the corresponding sequence

(b) forming an n-ary tree with leaf and interior nodes where:

(1) each leaf node represents a subset of values from one of said dictionaries, and

(2) each interior node associates a numeric index with tuples of numeric indexes from other terminal or non-terminal nodes.

7. The method of claim 6, whereby each unique value of a leaf node and each unique tuple of an interior node is associated with a count of the number of times that value or implied tuple of values occurred in the parallel data element sequences.

8. The method of claim 6, further including a means for efficiently processing a subset of a tree's leaves, comprising the following steps:

(a) the definition of a gate field in interior nodes,

(b) setting each of said gate fields' values, to indicate which of the corresponding interior node's branches lead to leaf nodes in said subset

(c) following paths that lead to said leaves, and

(d) processing the leaves encountered.

9. The method of claim 6, whereby an additional tree, t, is created using a subset of the same fields of the first tree, f, comprising the steps of:

(a) finding an ancestor node in tree f, of the leaf nodes in f corresponding to said subset of fields;

(b) looking up the tokens of said leaf nodes corresponding to a subset of tokens in said ancestor;

(c) inserting said leaf node tokens into tree, t.

10. The method of claim 6, further including the method for arranging said n-ary tree comprising the steps of:

(a) defining a problem space consisting of:

(1) a set of states such that each state contains a set of leaves and zero or more interior nodes, each with two or more other nodes as children,

(2) a value function, giving a numeric ranking of the value of any state's design

(b) defining one or more operators that transform one state to another, and

(c) searching the problem space, starting from an initial state and applying operators to move to other states until a state with an acceptable design is reached.

add Ap